Machine Learning for Data Science (CS4786)
Lecture 8

Kernel PCA
&
Isomap

Kernel Trick

- We have have nice methods for linear dimensionality reduction
- Can we use this beyond the linear realm?

Kernel Trick

- Lift to higher dimensions (introduces non-linearity)
  - Using feature map $\phi(x)$
- Perform linear dimensionality reduction in this high dimensional space

Key Idea:

If an algorithm only depends on inner products, we can simply replace inner product in x space by inner product in $\phi(x)$ space

Can we write PCA so it only depends on inner products?

Lets Rewrite PCA

Lets Rewrite PCA

Lets start with the assumption that Data is centered! (i.e. Sum of x_t's is 0)

Lets Rewrite PCA

Lets start with the assumption that Data is centered! (i.e. Sum of x_t's is 0)

- k^{th} column of W is eigenvector of covariance matrix
  That is, $\lambda_k W_k = \Sigma W_k$. Rewriting, for centered X

$$\lambda_k W_k = \frac{1}{n}\left(\sum_{t=1}^{n} \mathbf{x}_t \mathbf{x}_t^\top\right) W_k = \frac{1}{n}\sum_{t=1}^{n} \left(\mathbf{x}_t^\top W_k\right)\mathbf{x}_t$$

Lets Rewrite PCA

Lets start with the assumption that Data is centered! (i.e. Sum of x_t's is 0)

- k^{th} column of W is eigenvector of covariance matrix
  That is, $\lambda_k W_k = \Sigma W_k$. Rewriting, for centered X

$$\lambda_k W_k = \frac{1}{n}\left(\sum_{t=1}^{n} \mathbf{x}_t \mathbf{x}_t^\top\right) W_k = \frac{1}{n}\sum_{t=1}^{n} \left(\mathbf{x}_t^\top W_k\right)\mathbf{x}_t$$

But $\mathbf{x}_t^\top W_k = \mathbf{y}_t[k]$

Lets Rewrite PCA

Lets start with the assumption that Data is centered! (i.e. Sum of x_t's is 0)

- k^{th} column of W is eigenvector of covariance matrix
  That is, $\lambda_k W_k = \Sigma W_k$. Rewriting, for centered X

$$\lambda_k W_k = \frac{1}{n}\left(\sum_{t=1}^{n} \mathbf{x}_t \mathbf{x}_t^\top\right) W_k = \frac{1}{n}\sum_{t=1}^{n} \left(\mathbf{x}_t^\top W_k\right)\mathbf{x}_t$$

But $\mathbf{x}_t^\top W_k = \mathbf{y}_t[k]$

$$\lambda_k W_k = \frac{1}{n}\sum_{t=1}^{n} \mathbf{y}_t[k]\mathbf{x}_t$$

Lets Rewrite PCA

$$\mathbf{y}_s[k] = W_k^\top \mathbf{x}_s$$

Lets Rewrite PCA

$$\mathbf{y}_s[k] = W_k^\top \mathbf{x}_s$$

$$= \frac{1}{\lambda_k} \left(\frac{1}{n} \sum_{t=1}^{n} \mathbf{y}_t[k] \mathbf{x}_t \right)^\top \mathbf{x}_s$$

Lets Rewrite PCA

$$\mathbf{y}_s[k] = W_k^\top \mathbf{x}_s$$

$$= \frac{1}{\lambda_k} \left(\frac{1}{n} \sum_{t=1}^{n} \mathbf{y}_t[k] \mathbf{x}_t \right)^\top \mathbf{x}_s$$

$$= \frac{1}{n\lambda_k} \sum_{t=1}^{n} \mathbf{y}_t[k] \mathbf{x}_t^\top \mathbf{x}_s$$

Lets Rewrite PCA

$$\mathbf{y}_s[k] = W_k^\top \mathbf{x}_s$$

$$= \frac{1}{\lambda_k} \left(\frac{1}{n} \sum_{t=1}^{n} \mathbf{y}_t[k] \mathbf{x}_t \right)^\top \mathbf{x}_s$$

$$= \frac{1}{n\lambda_k} \sum_{t=1}^{n} \mathbf{y}_t[k] \mathbf{x}_t^\top \mathbf{x}_s$$

$$= \frac{1}{n\lambda_k} \sum_{t=1}^{n} \mathbf{y}_t[k] \tilde{K}_{s,t}$$

Lets Rewrite PCA

$$\begin{aligned}
\mathbf{y}_s[k] &= W_k^\top \mathbf{x}_s \\
&= \frac{1}{\lambda_k}\left(\frac{1}{n}\sum_{t=1}^{n}\mathbf{y}_t[k]\mathbf{x}_t\right)^\top \mathbf{x}_s \\
&= \frac{1}{n\lambda_k}\sum_{t=1}^{n}\mathbf{y}_t[k]\mathbf{x}_t^\top \mathbf{x}_s \\
&= \frac{1}{n\lambda_k}\sum_{t=1}^{n}\mathbf{y}_t[k]\tilde{K}_{s,t}
\end{aligned}$$

Where $\tilde{K}_{s,t} = \mathbf{x}_t^\top \mathbf{x}_s$ is the kernel matrix for centered data

Lets Rewrite PCA

- Hence, the k'th column on Y matrix is such that

$$\mathbf{y}[k] = \frac{1}{n\lambda_k}\mathbf{y}[k]\tilde{K}$$

Lets Rewrite PCA

- Hence, the k'th column on Y matrix is such that

$$\mathbf{y}[k] = \frac{1}{n\lambda_k}\mathbf{y}[k]\tilde{K}$$

$$\text{Also we have, } 1 = \|W_k\|^2 = \frac{1}{\lambda_k^2 n^2}\left(\sum_{t=1}^{n}\mathbf{y}_t[k]\mathbf{x}_t\right)^\top\left(\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s\right)$$

Lets Rewrite PCA

- Hence, the k'th column on Y matrix is such that

$$\mathbf{y}[k] = \frac{1}{n\lambda_k}\mathbf{y}[k]\tilde{K}$$

$$\text{Also we have, } 1 = \|W_k\|^2 = \frac{1}{\lambda_k^2 n^2}\left(\sum_{t=1}^{n}\mathbf{y}_t[k]\mathbf{x}_t\right)^\top\left(\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s\right)$$

$$= \frac{1}{\lambda_k^2 n^2}\sum_{t=1}^{n}\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s^\top\mathbf{x}_t\ \mathbf{y}_t[k]$$

Lets Rewrite PCA

- Hence, the k'th column on Y matrix is such that

$$\mathbf{y}[k] = \frac{1}{n\lambda_k}\mathbf{y}[k]\tilde{K}$$

$$\text{Also we have, } 1 = \|W_k\|^2 = \frac{1}{\lambda_k^2 n^2}\left(\sum_{t=1}^{n}\mathbf{y}_t[k]\mathbf{x}_t\right)^\top\left(\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s\right)$$

$$= \frac{1}{\lambda_k^2 n^2}\sum_{t=1}^{n}\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s^\top\mathbf{x}_t\ \mathbf{y}_t[k]$$

$$= \frac{1}{\lambda_k^2 n^2}\mathbf{y}[k]\tilde{K}\mathbf{y}[k]^\top$$

Lets Rewrite PCA

- Hence, the k'th column on Y matrix is such that

$$\mathbf{y}[k] = \frac{1}{n\lambda_k}\mathbf{y}[k]\tilde{K}$$

$$\text{Also we have, } 1 = \|W_k\|^2 = \frac{1}{\lambda_k^2 n^2}\left(\sum_{t=1}^{n}\mathbf{y}_t[k]\mathbf{x}_t\right)^\top\left(\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s\right)$$

$$= \frac{1}{\lambda_k^2 n^2}\sum_{t=1}^{n}\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s^\top\mathbf{x}_t\ \mathbf{y}_t[k]$$

$$= \frac{1}{\lambda_k^2 n^2}\mathbf{y}[k]\tilde{K}\mathbf{y}[k]^\top = \frac{1}{n\lambda_k}\|\mathbf{y}[k]\|^2$$

Lets Rewrite PCA

- Hence, the k'th column on Y matrix is such that

$$\mathbf{y}[k] = \frac{1}{n\lambda_k}\mathbf{y}[k]\tilde{K}$$

Also we have, $1 = \|W_k\|^2 = \frac{1}{\lambda_k^2 n^2}\left(\sum_{t=1}^{n}\mathbf{y}_t[k]\mathbf{x}_t\right)\left(\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s\right)$

$$= \frac{1}{\lambda_k^2 n^2}\sum_{t=1}^{n}\sum_{s=1}^{n}\mathbf{y}_s[k]\mathbf{x}_s^\top\mathbf{x}_t\ \mathbf{y}_t[k]$$

$$= \frac{1}{\lambda_k^2 n^2}\mathbf{y}[k]\tilde{K}\mathbf{y}[k]^\top = \frac{1}{n\lambda_k}\|\mathbf{y}[k]\|^2$$

Hence $P_k = \mathbf{y}[k]/\sqrt{n\lambda_k}$ is an eigenvector of $\tilde{K}$ with eigen value $\gamma_k = n\lambda_k$

Rewritting PCA

- We assumed centered data, what if its not,

Rewritting PCA

- We assumed centered data, what if its not,

$$\tilde{K}_{s,t} = \left(\mathbf{x}_t - \frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u\right)^{\top}\left(\mathbf{x}_s - \frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u\right)$$

Rewritting PCA

- We assumed centered data, what if its not,

$$\tilde{K}_{s,t} = \left(\mathbf{x}_t - \frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u)\right)^\top \left(\mathbf{x}_s - \frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u\right)$$

$$= \mathbf{x}_t^\top \mathbf{x}_s - \left(\frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u\right)^\top \mathbf{x}_s - \left(\frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u\right)^\top \mathbf{x}_t$$

$$+ \frac{1}{n^2}\left(\sum_{u=1}^{n}\mathbf{x}_u\right)^\top \left(\sum_{v=1}^{n}\mathbf{x}_v\right)$$

Rewritting PCA

- We assumed centered data, what if its not,

$$\tilde{K}_{s,t} = \left(\mathbf{x}_t - \frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u)\right)^\top \left(\mathbf{x}_s - \frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u\right)$$

$$= \mathbf{x}_t^\top \mathbf{x}_s - \left(\frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u\right)^\top \mathbf{x}_s - \left(\frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u\right)^\top \mathbf{x}_t$$

$$+ \frac{1}{n^2}\left(\sum_{u=1}^{n}\mathbf{x}_u\right)^\top \left(\sum_{v=1}^{n}\mathbf{x}_v\right)$$

$$= \mathbf{x}_t^\top \mathbf{x}_s - \frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u^\top \mathbf{x}_s - \frac{1}{n}\sum_{u=1}^{n}\mathbf{x}_u^\top \mathbf{x}_t + \frac{1}{n^2}\sum_{u=1}^{n}\sum_{v=1}^{n}\mathbf{x}_u^\top \mathbf{x}_v$$

Rewriting PCA

- Equivalently, if Kern is the matrix ($\text{Kern}_{t,s} = x_t^\top x_s$),

$$\tilde{K} = \text{Kern} - \frac{(\mathbf{1}_{n\times n} \times \text{Kern})}{n} - \frac{(\text{Kern} \times \mathbf{1}_{n\times n})}{n} + \frac{(\mathbf{1}_{n\times n} \times \text{Kern} \times \mathbf{1}_{n\times n})}{n^2}$$

Kernel PCA

Kernel PCA

1.

n Kern n $=$

$$\begin{bmatrix} k(x_1,x_1) & k(x_1,x_2) & \dots & k(x_1,x_n) \\ k(x_2,x_1) & k(x_2,x_2) & \dots & k(x_2,x_n) \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ k(x_{n-1},x_1) & k(x_{n-1},x_2) & \dots & k(x_{n-1},x_n) \\ k(x_n,x_1) & k(x_n,x_2) & \dots & k(x_n,x_n) \end{bmatrix}$$

Kernel PCA

1.

n Kern n $=$

$$\begin{bmatrix} k(x_1,x_1) & k(x_1,x_2) & \dots & k(x_1,x_n) \\ k(x_2,x_1) & k(x_2,x_2) & \dots & k(x_2,x_n) \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ k(x_{n-1},x_1) & k(x_{n-1},x_2) & \dots & k(x_{n-1},x_n) \\ k(x_n,x_1) & k(x_n,x_2) & \dots & k(x_n,x_n) \end{bmatrix}$$

2.

n $\tilde{K}$ n $= \text{Kern} - \frac{1}{n}(\mathbf{1}\ \text{Kern} + \text{Kern}\ \mathbf{1}) + \frac{1}{n^2}\mathbf{1}\ \text{Kern}\ \mathbf{1}$

Kernel PCA

Kernel PCA

$$3.\left[\underset{K}{\overset{n}{P}},\gamma\right] = \mathrm{eigs}\left(\tilde{K}, K\right)$$

Kernel PCA

3. $\left[\underset{K}{P}, \gamma \right] = \text{eigs}\left(\tilde{K}, K \right)$

4. $Y = \underset{K}{\begin{bmatrix} P_1\sqrt{\gamma_1} & \cdots & P_K\sqrt{\gamma_K} \end{bmatrix}}$

Kernel Methods: A note

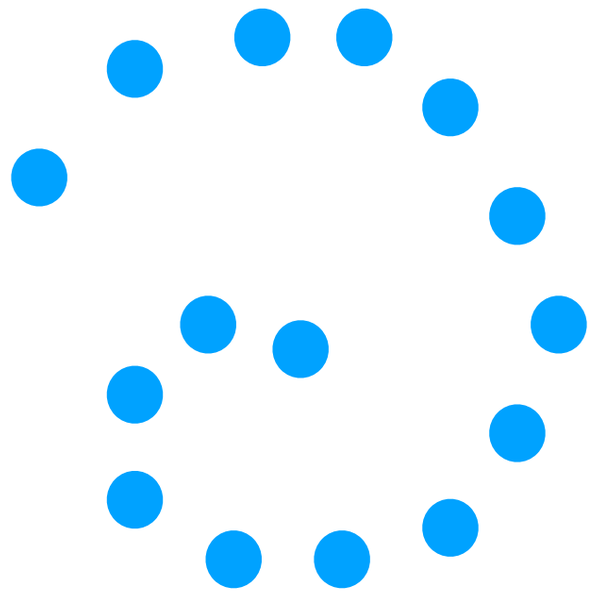
- We can “kernelize” CCA and many other linear dimensionality reduction methods.

- For any linear method, solution lies within linear span of data

- For typical linear methods y’s can be computed only based on inner products.

Manifold Based Dimensionality Reduction

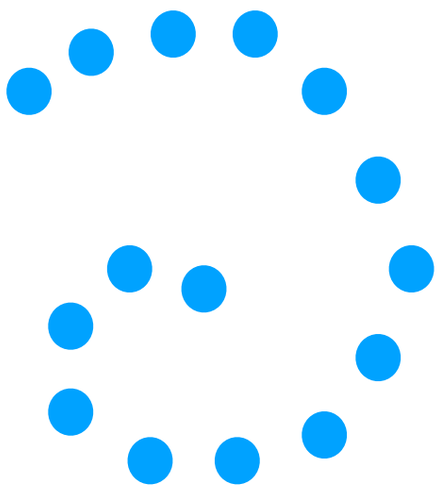
- Key Assumption: Points live on a low dimensional manifold
- Manifold: subspace that looks locally Euclidean
- Given data, can we uncover this manifold?



Can we unfold this?

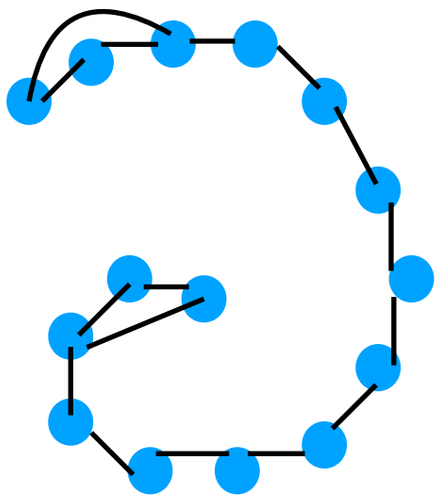
Method I: Isomap

1. For every point, find its (k-) Nearest Neighbors



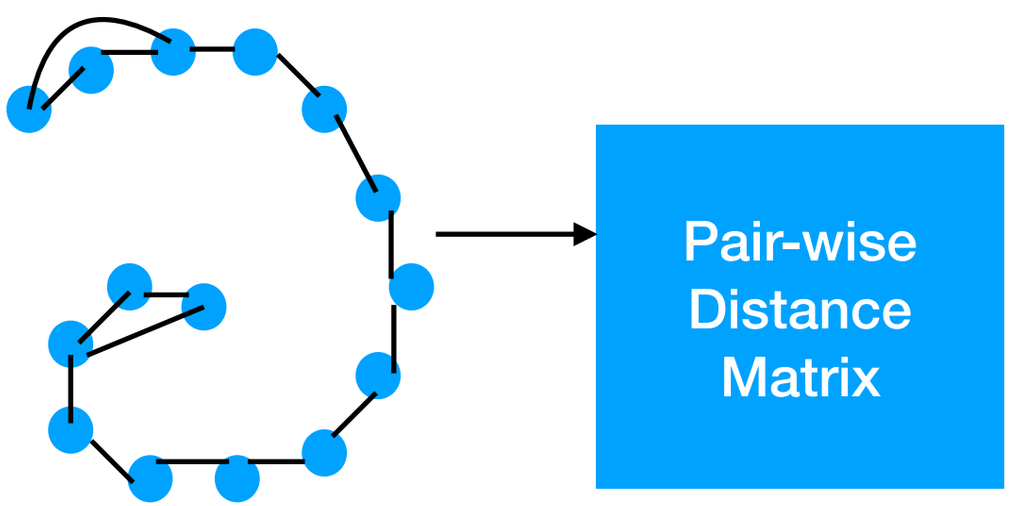
Method I: Isomap

1. For every point, find its (k-) Nearest Neighbors
2. Form the Nearest Neighbor graph



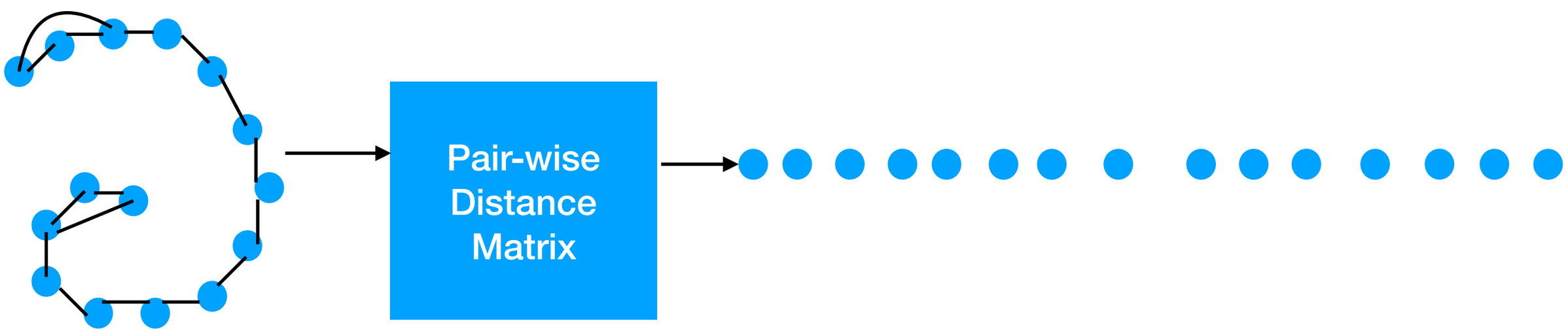
Method I: Isomap

1. For every point, find its (k-) Nearest Neighbors
2. Form the Nearest Neighbor graph
3. For every pair of points A and B, distance between point A to B is shortest distance between A and B on graph



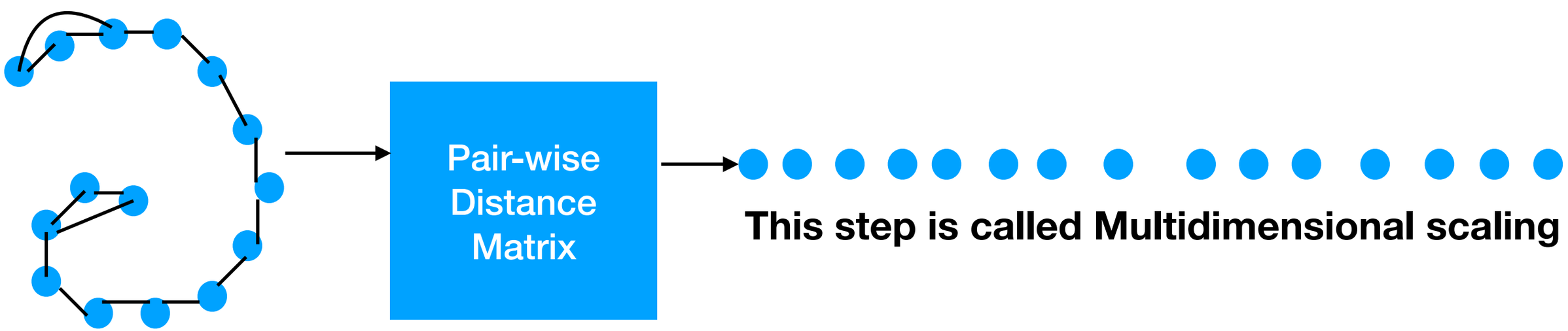
Method I: Isomap

1. For every point, find its (k-) Nearest Neighbors
2. Form the Nearest Neighbor graph
3. For every pair of points A and B, distance between point A to B is shortest distance between A and B on graph
4. Find points in low dimensional space such that distances between points in this space is equal to distance on graph.



Method I: Isomap

1. For every point, find its (k-) Nearest Neighbors
2. Form the Nearest Neighbor graph
3. For every pair of points A and B, distance between point A to B is shortest distance between A and B on graph
4. Find points in low dimensional space such that distances between points in this space is equal to distance on graph.



Isomap: Pitfalls

1. If we don't take enough nearest neighbors, then graph may not be connected

2. If we connect points too far away, points that should not be connected can get connected

3. There may not be a right number of nearest neighbors we should consider!

Demo